基于分级匹配的维吾尔语文档相似性计算及剽窃检测方法 *

亚森・艾则孜 [1]，艾山・吾买尔 [2a]，阿力木江・艾沙 [2b]

(1. 新疆警察学院 信息安全工程系，乌鲁木齐 830013; 2. 新疆大学 a. 信息科学与工程学院; b. 网络中心，乌鲁木齐 830046)

摘　要：针对以维吾尔语书写的文档间的相似性计算及剽窃检测问题，提出了一种基于内容的维吾尔语剽窃检测(U-PD)方法。首先，通过预处理阶段对维吾尔语文本进行分词、删除停止词、提取词干和同义词替换，其中提取词干是基于 N-gram 统计模型实现；然后，通过 BKDRhash 算法计算每个文本块的 hash 值并构建整个文档的 hash 指纹信息；最后，根据 hash 指纹信息，基于 RKR-GST 匹配算法在文档级、段落级和句子级将文档与文档库进行匹配，获得文档相似度，以此实现剽窃检测。通过在维吾尔语文档中的实验评估表明，提出的方法能够准确检测出剽窃文档，具有可行性和有效性。

关键词：维吾尔语文档；相似度；剽窃检测；文档 hash 指纹；分级匹配

中图分类号：TP391.1　　doi: 10.3969/j.issn.1001-3695.2017.12.0853

Uyghur document similarity calculation and plagiarism detection based on hierarchical matching

Yasen·aizezi[1], Aishan·wumaier[2a], Alimu·aisha[2b]

(1. *Dept. of Information Security Engineering Xinjiang Police College*, *Urumqi* 830011, *China*; 2. *a. SchoolofInformation & Engineering*, *b. Network Center*, *Xinjiang University*, *Urumqi* 830046, *China*)

Abstract: For the issues of the similarity calculation and plagiarism detection from documents written in Uyghur, a content-based Uyghur plagiarism detection (U-PD) method is proposed. Firstly, the Uyghur texts are segmented, the stop words are deleted, the stems are extracted and synonyms are replaced through the preprocessing stage, of which extraction stems are based on N-gram statistical models. Then, calculate the hash value of each text block through the BKDRhash algorithm and construct the hash fingerprint information of the entire document. Finally, according to the hash fingerprint information, the document and document library are matched at the document level, the paragraph level and the sentence level based on the RKR-GST matching algorithm, and the similarity of the document is obtained, so as to realize plagiarism detection. The experimental evaluation in Uyghur documents shows that the proposed method can detect plagiarism documents accurately and is feasible and effective.

Key words: Uyghur documents; similarity; plagiarism detection; document hash fingerprinting; hierarchical matching

0　引言

通过网络获取信息十分容易，这使学术剽窃成为一种简单操作。存在以下几种类型的文档剽窃[1]：a)直接从发表的文本中复制短语或段落，而不给出引用出处和作者；b)将已发表内容进行语句和结构修改并进行使用。为了保护作者的版权，对待发表文档进行剽窃检测是一种重要手段[2]。近些年，随着国家对新疆地区经济和教育发展的大力支持，产生了很多以维吾尔语进行书写的学术论文[3]。对维语文档进行相似度计算和剽窃检测对维语文化的健康发展具有重要意义。

由于文本中的每个单词都可能有很多同义词，且具有不同的含义，给剽窃检测造成了一定的难度[4]。传统的剽窃检测方法主要是手动的，将文本相互比较以检测复制粘贴内容。传统方法易于应用，但通常需要较长的处理时间，并且不可靠，特别是在大文本的情况下。为此，需要自动工具来帮助用户快速准确地检测剽窃。常用的自动剽窃检测方法可分为与内容无关

收稿日期：2017-12-18；**修回日期：**2018-03-09　**基金项目：**国家自然科学基金资助项目（61762086，61662077，61363064）；国家社会科学基金资助项目（13CFX055）；新疆维吾尔自治区高校科研计划项目（XJEDU2016I052，XJEDU2017M046）；

作者简介：亚森・艾则孜（1975-），男（维吾尔族），新疆库车人，教授，国家电子数据司法鉴定员，硕士，主要研究方向为数字取证、自然语言处理等；艾山・吾买尔（1981-），男（维吾尔族，通信作者），新疆库车人，副教授，博士，主要研究方向为自然语言处理（yasenaizezi@126.com）；阿力木・艾沙（1973-），男（维吾尔族），新疆喀什人，副教授，博士，主要研究方向为人工智能.

的检测方法和与内容相关的检测方法[5]。与内容无关的方法是基于评估特定语言中的固有文本特征，例如单个字符的数量和句子的平均长度。与内容相关的方法是基于评估特定词语的特征。例如，特殊词语的频率属性。基于内容的方法依赖于文档内容的具体表示。文档指纹[6]是一种表示文档的有效技术，可通过比较文档的指纹信息来测量两个文档的相似性。文档指纹是由文档的 hash 子集创建的一组整数构成，用来表示文档的关键内容。

由于维吾尔语文档的信息化处理发展较晚，目前在维吾尔语文档相似性、文档过滤等方面的研究单位主要为新疆大学。常用的英文文档相似性度量有词频-逆文档频率(TF-IDF)、　信息增益、互信息和余弦相似度。由于维吾尔语的复杂语言结构，这些度量都不能很好地应用，为此检测维吾尔语文档中的剽窃是一项具有挑战性的任务。由于维吾尔语形态变化多样，如何提取用于表示一篇文档的关键词或语言模型，以此来计算文档间的相似性是一个难点。目前很少有学者提出相关方法，其中文献[7]提出一种维吾尔语句子相似度计算方法(MUSM)，其采用词形特征，通过多策略精选算法来计算两个维吾尔语句子的相似度。然而，其只能在句子级进行检测，且没有考虑到同义词的替换问题。文献[8]首先引入和分析了维吾尔语文本语义相似性度量，通过上下文来确定语义相似度。但是其精确性较低，不能用来进行剽窃检测。

本文提出了一种基于内容的维吾尔语剽窃检测(Uygur - plagiarism detection, U-PD)方法。其主要工作如下：a)通过分词、删除停止词、n-gram 词干提取和同义词替换操作实现文档的预处理，以此使后续相似度计算能够对词形变换和同义词具有鲁棒性；b)通过 hash 指纹信息来表示文本，以此提高文本表示的准确性和计算效率，其是通过 BKDRhash 算法实现；c)通过 RKR-GST 匹配算法在文档级、段落级和句子级对文档进行匹配，以此实现剽窃检测。实验结果证明了提出方法的有效性。

1　维吾尔语特征

维吾尔语是一种高度黏着性语言，其单词由 32 个字母组成，每种字母有 4 种不同的形式，致使其时态和形态变化比英语更丰富。维吾尔语中，通过在单词的结尾添加不同的词缀来实现语法功能。即很多词语是由同一词干演变而来的，且这些单词的词义相差不大[9]。由于这些特征，导致维吾尔语文本的原始特征维数大、文本表示稀疏等问题[10]，和传统中文或英文的文本相差很大。

维吾尔语的动词和一部分名词是由词干中形成的。词汇具有固定模式，通过在词的前后添加前缀和后缀可以表示它的数、性和时态。表 1 展示了可能添加到单词“شائىر”（诗人）中的不同词缀及其含义，其中，下划线上的字母为词缀。

2　维吾尔语文本剽窃检测方法框架

自然语言剽窃检测系统应满足以下特性：对标点符号、空格、大小写等不敏感；对较少的匹配不敏感(匹配应该足够大，才意味着剽窃)；对文件内容的置换不敏感。

本文提出的维吾尔文剽窃检测工具 U-PD 是基于内容的方法构建的。U-PD 的主要架构如图 1 所示。其主要步骤有：a)预处理过程，包含分词、停止词删除、生成词干和同义词替换;b)hash 指纹表示过程，即使用 Hash 指纹数据来表示文档信息;c)相似性检测，将文档构建成三层树结构，使用匹配算法来找到各级中两个 hash 字符串的最长匹配。

表 1　词干“شائىر”（诗人）上的添加不同词缀形成的单词

维吾尔语单词	词义	维吾尔语单词	词义
شائىر	诗人	شائىردا	在诗人
شائىرە	诗人（女）	شائىرەدە	在诗人（女）
شائىرنىڭ	诗人的	شائىردەك	像个诗人
شائىرلار	诗人们	شائىرىم	我的诗人
شائىرلەر	诗人们（女）	شائىرىڭ	你的诗人
شائىرلارنىڭ	诗人们的	شائىرەسى	他的诗人（女）

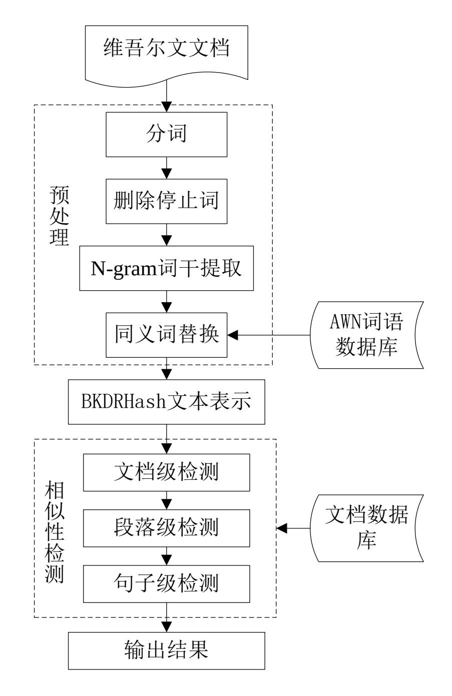


图 1　提出的 U-PD 方法主要框架

3　文档预处理

3.1　预处理基本步骤

大多数基于内容的检测方法都需要一个预处理阶段，用来删除停止词并生成词干，最终将维吾尔语文本转换为结构化表示，有利于后续的剽窃检测过程。预处理过程具体描述如下：

a)分词。将输入文本分解成字符。

b)删除停止词(SW)。因为停止词在任何文本中被视为文档间不重要的差异。将其删除可有效减低文本表示维度，并减少误报数，以获得更显著的结果。

c)生成词干(Stem)。通过删除最长的后缀和前缀，然后通过

N-gram 统计模型计算单词与词库词干的相似性，以此将单词缩减为其词干。

d)同义词替换(Synonym)。将词转换为最常见的同义词有助于检测隐藏剽窃的高级形式。词汇同义词从维吾尔语 WordNet 中检索，以同义词列表中的第一个同义词认为是最常见的单词。

3.2　基于 n-gram 统计模型的词干提取

对于词干提取，通常有基于词干的方法和基于统计的方法，相比而言，基于统计的方法更适合维吾尔语文本分类任务。本文采用了 n-gram 统计模型[11]来提取维吾尔语词干。采用的 n-gram 为字母级别，将所有连续的 N 个字母序列作为一个单元，称为一个 gram。

N-gram 模型中，其设定一个字母单元 l_i 在文本中出现的概率只与前 N-1 个字母相关。因此，字母序列 $L=l_1l_2l_3,\ldots,l_N$ 出现的概率表示为：

$$P(L)=P(l_1l_2l_3,\ldots,l_N)=\prod_{i=1}^{N}P(l_i\mid l_{i-N+1},\ldots,l_{i-1}) \tag{1}$$

在维吾尔语中，由于字母相互结合的概率很高，所以较小的 N 不能很好地表现单词属性，而较大的 N 可能会过滤掉一些有意义的词。根据维吾尔语的结构特征，设置 n=3、4 时具有较强的代表性。通过实验分析，当 n=3 时效果最好。为此，本文在实验部分设定 n=3。

在基于 n-gram 统计模型的词干提取方法中，首先移除了单词中最常见的前缀和后缀，也包含外国语、数字、停止词等。然后，通过 n-gram 模型计算两个词的相似性，以此来提取词干。基于 n-gram 统计模型的词干提取算法如算法 1 所示。

```
算法 1：基于 N-gram 统计模型的词干提取算法
for 文本中的每个词
  if 非维吾尔语词汇
      then 该词是无用词；
  if 包含数字
      then 该词是无用词；
  if 单词长度<3
       then 该词是无用词；
  移除附加符号，并标准化词汇；
  if 该词是停止词
        then 该词是无用词；
  移除前缀和后缀；
  if 该词是停止词
        then 该词是无用词；
  利用 n-gram 统计模型计算单词间相似性获得词干；
end for
```

首先，算法确保单词是一个维吾尔语词，并认为长度少于 3 个字母的词在文章中是不重要的。接着会移除各种附加符号，这些符号在字母的上面或下面用于正字法，作为词法的标志。之后应用词标准化方法，将一些字母的不同写法统一为相同的形式，如：将ب,بـ,ـبـ,ـب统一为ب ；将ھ,ە 统一为ھ；将چ,چـ,ـچـ,ـچ统一为چ等。

词形标准化后，算法会检查单词是否在一个停止词表中。停止词表由 165 个单词组成。消除停止词后，算法移除一组前缀(تاپ، قاپ، سۈپ،ئاي، مر، نا، بى 等)。移除后，算法会检查单词长度是否小于 3 个字母，如果小于 3 个，说明前缀是单词的一个主要部分，因此移除的前缀会恢复到单词中。接着将后缀(نىڭ،دا،تا،دىن،تىن، داش،چى،خان،غان،گەن،لار،لەر،ى،مى 等)递归地从词尾移除。首先从最长的后缀开始，再移除较短的。当词的前缀和后缀都移除之后，算法还会检查该词是否属于停止词表中的词汇，这是因为一些停止词也会附加前缀和后缀。

最后，利用 N-gram 统计模型计算单词间的相似性获得最终词干。对语料库中的所有术语对，计算其相似性度量。具有高于预定义相似性阈值的术语被聚类，并仅用其中一个术语来表示。

下面的例子描述了基于 N-gram 模型(N=2)，计算两个词 سىياسەت（政治）和 سىياسىنىڭ （政治的）的相似性。

a)سىياسەت⇒ سى، يا، سە ، ت 。（首先将词分解为两字母组合模型）

b)分解成的两字母组合 ⇒ سى، يا، سە ، ت。

c سىياسىنىڭ ⇒ سى ، يا ، سى ، نى ،ڭ 。

d)分解成的两字母组合 ⇒ سى ، يا ، سى ، نى ،ڭ 。

那么相似性为 $S=\dfrac{2C}{A+B}=\dfrac{2\times3}{4+3}=0.8571$。其中，$A$ 和 B 分别表示第一个词和第二个词中不同的两字母组合数量，C 表示两个词共同的两字母组合数量。将相似性大于阈值 T_S 的两个词归为一个词干。

图 2 展示了 U-PD 方法中一个维吾尔语文本句子预处理的例子。

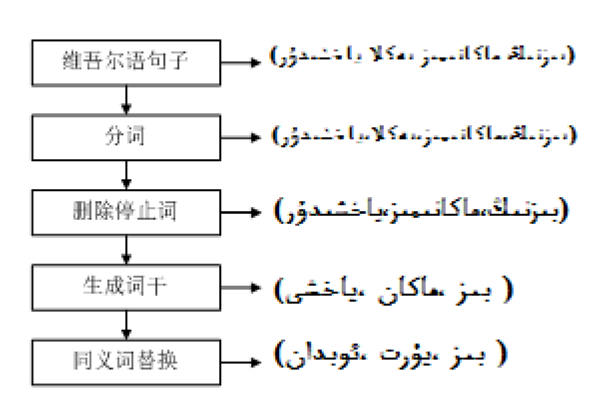


图 2　一个维吾尔语文本预处理的例子

4　基于 hash 指纹的文本表示

数字指纹[12]是一种散列(hash)函数，用来将文本映射到另一个文本上。另外，数字指纹可以把文本压缩成摘要，减少数据量并规范化其格式。hash 指纹作为文本数据的映射，对应着不同的文本数据。在文本相似性计算过程中，只需要根据文本数据的 hash 指纹信息，通过比较来判断相似性。

生成 hash 指纹的技术包括 i-hash 方法、0 mod p hash 方法以及 Winnowing 方法[13]。在 i-hash 方案中，选择每个文档的第 i 个 hash 值。该方法易于实现，但在存在文本插入、删除或重新排序的情况下不可靠。例如，如果将一个字母插入到文本中，则指纹将被移位一个，这改变了原始文档并且没有共享指纹，因此不会检测出到副本。在 0 mod p 方案中，其中 p 是整数，选择位于每个 0 mod p 处的 hash 值，使所有的 hash 值中的 1/p 被保留下来作为文档指纹。通过统计文档中相同指纹的数量来检测文档相似性。但其不能保证文档间的所有匹配会被检测出来。Winnowing 算法是一种本地指纹识别算法，其使用了滑动窗从 hash 序列中选择合适指纹。令 t 和 k 分别为保证阈值和噪声阈值。必须满足以下两个属性才能确定两个文档之间的匹配：(1)如果存在与保证阈值 t 一样长的匹配子串，则检测到匹配；(2)没有检测到比噪声阈值 k 短的任何匹配。Winnowing 算法包括以下步骤：首先给定 $t-k+1$ 的窗口大小，每个窗口 w_i 包含 hash 值 $h_i \dots h_{i+w-1}$。然后从每个窗口中选择最小 hash 值作为指纹。如果存在多个具有最小值的 hash，则选择最右边的一个。最后将所有选定的 hash 作为文档指纹。但是这些方法都不能很好的表示文档属性，为此本文先将文档进行分块，并从这些块中通过 BKDRhash 算法来获得各块的 hash 值作为其表示。

为了提取文档的 hash 指纹，本文首先需要将文本切成较小的块。一个句子或一个单词可以用作一个单元块。在基于句子的分块中，基于块参数 n 将文档中连续 n 个句子分组成一个块。例如，给定一个包含句子 $s1s2s3s4s5$ 的文档，如果 $n=3$，那么这些块是 $s1s2s3$、$s2s3s4$ 和 $s3s4s5$。在基于单词的分块中，文档根据块参数 n 将连续 n 个单词分组成一个块。例如，给定包含 $w1w2w3w4w5$ 的文档，如果 $n=3$，那么这些块是 $w1w2w3$、$w2w3w4$ 和 $w3w4w5$。基于词的分块比基于句子的分块能够为相似度检测提供更高的精度。本文 U-PD 采用了基于单词的分块方法，然后计算每个块的 hash 值。

选择一个 hash 函数是很重要的，因为将不同的块映射到相同的 hash 可以最小化冲突。例如，用于将每个块映射到块字符整数值之和的 hash 函数是很容易实现。但是，这不是一个精确的 hash 函数，因为具有不同顺序的相同字符的块具有相同的 hash 值(冲突)。在本文中，使用 BKDRhash (来自 Brian Kernighan 和 Dennis Ritchie)函数来进行分块。此函数通过一个特殊值(命名为 seed，通常等于 31)，返回每个字符的乘法之和。seed 值应该是一个素数，以保证 hash 值的唯一性。BKDRhash 算法描述算法 2 所示。

算法 2：BKDRhash 算法

```
BKDRhash（str）
{
seed =31, hash =0;
for（i=0; i< str.length; i++）
{
hash=hash*seed + str.charAt(i);
}
return（hash&0x7FFFFFF）;
}
```

在 BKDRhash 算法 hash 值计算过程中，对于长度为 k 的字符串 $c_1c_2\cdots c_k$，设定一个基数 b 和一个素数 q，那么该字符串的 hash 值表示为

$$Hash(c_1c_2\cdots c_k)=\begin{pmatrix} ask(c_1)\times b^{k-1}+ask(c_2)\times b^{k-2} \\ +\cdots+ask(c_{k-1})\times b+ask(c_k) \end{pmatrix} \bmod q \quad (1)$$

下一组字符串 $c_2c_3\cdots c_{k+1}$ 的 hash 值表示为：

$$Hash(c_2c_3\cdots c_{k+1})=\begin{pmatrix} Hash(c_1c_2\cdots c_k)- \\ ask(c_1)\times b^{k-1}\times b+ask(c_{k+1}) \end{pmatrix} \bmod q \quad (2)$$

本文选择 BKDRhash 算法来生成文档 hash 指纹的原因如下：1）该算法不需要考虑输入数据类型，只需要将其作为字符串来进行计算即可；2）在相似度计算过程中只需要比较 hash 值，有效减少了计算量，提高了检测效率。

5 基于分级比较的文本相似性检测

5.1 相似性度量

hash 指纹比较的相似性指标有很多种，包括 Levenshtein 距离、最长公共子串(longest common subsequences，LCS)、Karp-Rabin 贪心字符串匹配(running Karp-Rabin greedy string tiling，RKR-GST)[14]。Levenshtein 距离测量最小操作数：插入、删除或替换，以及将一个字符串转换为另一个字符串。例如，“Saturday”和“Sunday”之间的 Levenshtein 距离是 3。LCS 用来找到两个字符串中通用的最长子串。例如，“Saturday”和“Sunday”中常见的最长子串是“day”。RKR-GST 是一种加速 GST 算法的改进技术，为模式字符串中长度为 s 的每个子字符串和文本字符串中长度为 s 的每个子字符串创建一个 hash 值，并将模式字符串中的每一个 hash 值与文本字符串中的 hash 值进行比较。如果模式和文本 hash 值相等，则表明该模式与相应文本子串之间存在匹配。

相似度检测的一个关键问题是选择适当的度量。对于剽窃检测，RKR-GST 和 LCS 更适合，因为剽窃涉及对文本的修改(插入，删除等)。在本文 U-PD 中，由于通过 hash 指纹函数将文本通过 Hash 值进行表示，为此采用 RKR-GST 算法更加适合。

5.2 RKR-GST 算法

GST 是一种贪心串匹配算法，用来检测两个字符串是否完全相等。KR 算法是一种随机串匹配算法，用于在文本串中找出匹配模式串出现的位置。RKR-GST 算法结合了 GST 算法与 KR 算法各自的优点，只有当模式子串与文本子串的 hash 值相同时才需要进行比较，为此不需要逐一比较模式串与文本串中的每个字符，有效提高了匹配过程的运行效率[15]。

RKR-GST 算法中，首先设定一个长度参数 s，将两个字符串 T 与 P 划分为各个小块，并计算各小块的 hash 值。然后执行一个迭代扫描过程，对这两个字符串的 hash 值进行贪婪匹配，

直到不能匹配为止。记录两个字符串 T 与 P 的最大匹配长度 MatehLength 和匹配的开始位置，形成一个匹配列表。接着，执行一个标记过程，判断字符串匹配是否发生重叠，如果重叠则删除。最后，改变长度参数 s，重复进行扫描过程和标记过程，直到长度参数小于最小匹配长度 Min-Length，算法停止。RKR-GST 算法的流程如图 3 所示。

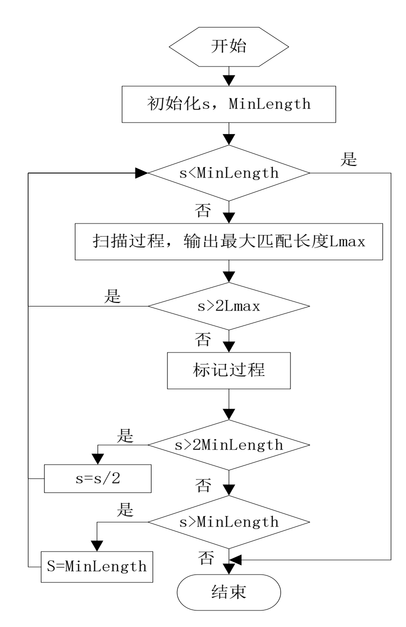


图 3 RKR-GST 算法流程

传统 GST 算法能够应对字符串中改变字符顺序的问题，在剽窃检测中应用较多，但其需要比较两个字符串中的每个元素，时间复杂度较大。其最坏情况下的时间复杂度为 $O(n^3)$，最好情况下的时间复杂度为 $O(n^2)$，n 为字符串长度。而 RKR-GST 算法中，只有当子串的 hash 值相等时才需要进行比较，进行匹配的子串数量较少，提高了匹配效率。其最好情况下的时间复杂度可以降低到 $O(n)$，最坏情况下的时间复杂度为 $O(n^3)$。

5.3 基于树结构的相似性检测

为每个文档创建树结构表示，以描述其逻辑结构，如图 4 所示。树根表示文档本身，第二级代表段落，叶节点表示句子。这种表示法旨在避免多个文档之间不必要的比较。然后从上到下遍历树，并先后在文档级别、段落级别和句子级别进行比较。

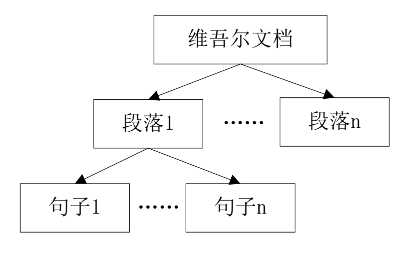


图 4 文档的树结构表示

为树的每个级别采用了一种比较算法：算法 3(文档级)、算法 4(段落级)和算法 5(句子级)。

在文档级别，根据两个文档的 hash 值和固定阈值对两个文档进行比较。如果交集中的 hash 值大于阈值，则两个文档之间存在潜在的相似性。在这种情况下，将在段落级别上继续比较过程，否则没有检测到相似性，并停止比较。如果在段级别检测到可能的相似性，则在句子级继续比较过程，否则终止该过程。如果两个句子之间存在可能的相似性，则使用 RKR-GST 算法进行相似性测量。如果最长匹配长度大于最小句子的长度与阈值的乘积，则在两个句子中标识相似的字符串，然后继续下一句的比较。文档相似度的表达式为：

$$Doc_{similarity} = \frac{DocSize_{Intersect}}{DocSize_{Min}} \tag{3}$$

其中: $DocSize_{Min}$ 表示两个文档的最小尺寸， $DocSize_{Intersect}$ 表示两个文档的交集尺寸，即 RKR-GST 算法获得的最大匹配长度。段落和句子的相似度计算同理。

算法 3：文档级比较算法
输入：文档 A 和 B (两个输入文档)
输出：文档相似性
开始
最小文档尺寸 $DocSize_{Min} = \min(|A|,|B|)$
文档交集尺寸 $DocSize_{Intersect} = \min(|A| \cap |B|)$
如果($DocSize_{Intersect} \geq DocSize_{Min} * Doc_{Threshold}$)
 那么 $similarity = true$
 否则 $similarity = false$
结束

算法 4：段落级比较算法
输入：段落 A 和 B (两个输入段落)
输出：段落相似性
开始
段落最小尺寸 $ParSize_{Min} = \min(|A|,|B|)$
段落交集尺寸 $ParSize_{Intersect} = \min(|A| \cap |B|)$
如果($ParSize_{Intersect} \geq ParSize_{Min} * Par_{Threshold}$)
 那么 $similarity = true$
 否则 $similarity = false$
结束

算法 5：句子级比较算法
输入：句子 A 和 B (两个输入句子)
输出：句子相似性
开始
句子最小尺寸 $SenSize_{Min} = \min(|A|,|B|)$
句子交集尺寸 $SenSize_{Intersect} = \min(|A| \cap |B|)$
如果($SenSize_{Intersect} \geq SenSize_{Min} * Sen_{Threshold}$)
那么 $similarity = true$
 否者 $similarity = false$
结束

6 实验及分析

6.1 实验设置

利用 Java 语言实现了 U-PD 算法，并在一个包含已手工标

记的 300 个维吾尔语文档的数据测试集中评估了其性能，每个文档大约有 800 个字。从原始文档中生成 3 个具有同义词和结构变化的数据集，来评估 U-PD 在检测隐藏剽窃方面的性能。数据集表示如下：

数据集 1：同义词。选择 100 个文档，随机选取文档中 50％的单词，并用其中一个同义词替换这些单词。

数据集 2：结构变化。选择 100 个文档，随机选取文档中 50％的句子，并改变这些句子的结构。

数据集 3：同义词+结构变化。选择 100 个文档，随机选取文档中 40％的句子，改变这些句子的结构，并用同义词替换其中 20%的单词。

对于相似性检测系统中的参数，通过多次实验结果分析来选择合适参数。最终设定 BKDRhash 函数分块中的块参数为3。文档相似阈值 $Doc_{Threshold}$ 设置为 0.1，即描述不同主题的文档的交叉点数量小于最小文档大小的 10％。段落相似性阈值 $Par_{Threshold}$ 设置为 0.2；句子阈值 $Sen_{Threshold}$ 设置为 0.5。由于本文算法中，当文档相似性大于 $Doc_{Threshold}$ 时，才会进行段落相似性检测。为此，本文设置 $Doc_{Threshold}$ 较小，为 10%，以免漏掉一些相似文档。另外，当段落相似性大于 $Par_{Threshold}$ 时，才会进行句子相似性检测，为了进一步筛选过滤，为此设置 $Par_{Threshold}$ 大于 $Doc_{Threshold}$，为 20%。在最后的句子级检测中，为了综合考虑漏检和错检，设置当两个句子间的相似性超过 50%时才判断为相似。

采用召回率(Recall) [16]和精度(Precision)指标来度量检测性能，表示如下：

$$Recall = \frac{\text{确定的剽窃单元数}}{\text{剽窃单元总数}} \times 100\% \tag{4}$$

$$Precision = \frac{\text{确定的剽窃单元数}}{\text{确定的单元总数}} \times 100\% \tag{5}$$

6.2 预处理过程性能分析

U-PD 预处理过程包括停止词删除，生成词干和同义词替换步骤，为了评估这些步骤对算法性能的影响，构建了 3 种预处理策略，即只有停止词删除(SW)；停止词删除+生成词干(SW+Stem)；停止词删除＋生成词干＋同义词替换(SW+Stem+Synonym)。将 3 种预处理后的文档输入到相似性检测系统中进行检测。

图 5 和 6 分别显示了具备三种预处理策略的 U-PD 算法在三个数据集上得到的平均精度 $Mean(Precision)$ 和平均召回率 $Mean(Recall)$。

可以看出，只具备 SW 预处理过程的检测算法没有检测到隐藏的剽窃(同义词替换和结构变化)。其在所有数据集上的整体表现都较差，其中在数据集 3 上的结果为 $Mean(Precision) = 53.7\%$。$Mean(Recall) = 45.6\%$。

具备 SW+Stem 预处理过程的检测算法没有检测到同义词替换，但它可以高精度地识别更改句子结构的相似性文档，其中在数据集 3 上的结果为 $Mean(Precision) = 83.6\%$，$Mean(Recall) = 72.8\%$。这表明缩减单词到其词干可以增强剽窃检测的性能。

具备 SW+Stem+Synonym 预处理过程的检测算法效果最好，在数据集 3 上的结果为 $Mean(Precision) = 96.5\%$，$Mean(Recall) = 94.6\%$。其中，以 $Mean(Precision) = 97.6\%$ 检测出了同义词替换，以 $Mean(Precision) = 93.2\%$ 检测出了结构变化。

从上述实验可以得出结论，本文包括停止词移除、生成词干和同义词替换步骤的预处理过程对后续相似性检测有很大的促进作用。其中，基于 N-gram 统计模型的词干提取过程能够很好地应对剽窃文档中改变句子结构的情景。同义词替换过程能够很好地应对剽窃文档中的同义词替换的情景。为隐藏剽窃行为的有效检测提供了良好的输入特征。

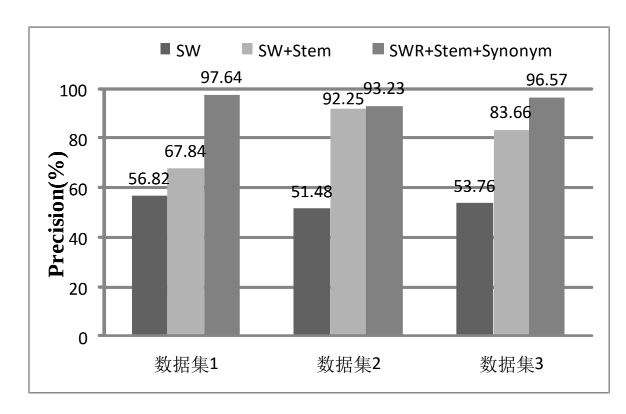


图 5　具备各种预处理方法的算法在数据集上的 Precision（%）

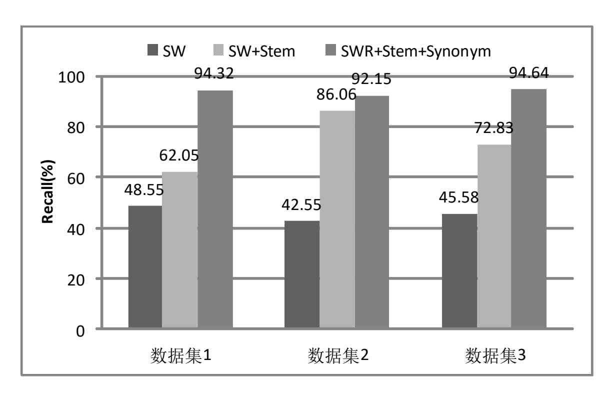


图 6　具备各种预处理方法的算法在数据集上的 Recall（%）

6.3 性能比较

将本文提出的 U-PD 方法(预处理过程包含 SW+Stem+Synonym)与文献[7]提出的基于词形特征的多策略精选维吾尔语句子相似度计算方法(MUSM)进行比较。表 2 显示了在每个数据集上由 U-PD 和 MUSM 给出的平均精度和召回率，其中每个实验执行了 5 次。为了表示各种方法的稳定性，还分别计算了精度和召回率结果的标准差 σ。

表 2 可以看出，由于 MUSM 无法检测到任何同义词替换，所以其性能接近 U-PD 方法(预处理过程包含 SW+Stem)的检测结果。总的来说，U-PD 性能优于 MUSM。对于 U-PD，3 个数据集上的总体 $Mean(Precision) = 95.8\%$。

$Mean(Recall) = 93.7\%$ 。对于 MUSM，3 个数据集上的总体 $Mean(Precision) = 90.2\%$ ， $Mean(Recall) = 87.4\%$ 。另外，可以看出 U-PD 方法的稳定性较好，其 $\sigma(Precision) = 2.62\%$ ， $\sigma(Recall) = 3.35\%$ 。而 MUSM 方法的稳定性较差，同一个数据集上不同次实验的结果相差较大，其 $\sigma(Precision) = 5.45\%$ ， $\sigma(Recall) = 5.97\%$ 。

表 2　本文 U-PD(SW+Stem+Synonym)和 MUSM 的结果比较

数据集	性能	U-PD	MUSM
数据集 1	$Mean(Precision)$	97.64%	96.85%
	$Mean(Recall)$	94.32%	94.18%
数据集 2	$Mean(Precision)$	93.23%	93.31%
	$Mean(Recall)$	92.15%	91.96%
数据集 3	$Mean(Precision)$	96.57%	80.64%
	$Mean(Recall)$	94.64%	76.09%
总体平均	$Mean(Precision)$	95.81%	90.27%
	$Mean(Recall)$	93.70%	87.41%
	$\sigma(Precision)$	2.62%	5.45%
	$\sigma(Recall)$	3.35%	5.97%

7　结束语

该文提出了一种维吾尔语文档剽窃检测方法，可以检测到一些隐藏的剽窃形式，例如句子结构变化和同义词替换。通过预处理阶段来提取词干并转换成指定的同义词。然后通过 BKDRhash 算法将文档以 hash 指纹信息来表示。最后基于 RKR-GST 匹配算法在文档级、段落级和句子级对文档进行匹配，实现剽窃检测。通过一系列实验证明了本文方法对大量维吾尔语文档的有效性，能够精确检测出存在复制、句子结构变化和同义词替换的剽窃文档。

在今后工作中，将考虑其他同义词数据库，并进一步优化方法中的参数(如阈值和块值)，进一步提高本文方法的检测准确性。

参考文献：

[1] 邹杜，陈育青，张凌. 基于语义匹配的抄袭检测方法 [J]. 华南理工大学学报：自然科学版, 2013, 41 (7): 131-136.

[2] 张超，陈利，李琼. 一种 PST_LDA 中文文本相似度计算方法 [J]. 计算机应用研究, 2016, 33 (2): 375-377.

[3] 吐尔地·托合提，维尼拉·木沙江，艾斯卡尔·艾木都拉. 基于语义串抽取及主题相似度度量的维吾尔文文本分类 [J]. 中文信息学报, 2017, 31 (4): 100-107.

[4] Sindhu L, Idicula S M. A plagiarism detection system for malayalam text based documents with full and partial copy [J]. Procedia Technology, 2016, 25 (4): 372-377.

[5] Bin Wang, Xiaochun Yang, Guoren Wang. Copy detection among programs using extreme learning machines [J]. 2015, 4 (6): 189-198.

[6] 张祯，邓新洁，付二帅，等. 一种用于文本复制检测的指纹特征选择算法 [J]. 杭州电子科技大学学报, 2017, 37 (3): 51-57.

[7] 田生伟，吐尔根•依布拉音，禹龙，等. 一种维吾尔语句子相似度算法的研究 [J]. 计算机工程与应用, 2009, 45 (26): 144-146.

[8] Ma B, Zhou X, Yang Y, *et al.* Uyghur semantic similarity computation based on contextual information in web documents [J]. Journal of Computational Information Systems, 2012, 8 (2): 563-570.

[9] 买买提依明•哈斯木，吾守尔•斯拉木，维尼拉•木沙江，等. 基于 N 元模型的维吾尔语文本分类技术研究 [J]. 计算机应用研究, 2015, 32 (7): 1986-1988.

[10] Mi C, Yang Y, Wang L, *et al.* Detection of loan words in uyghur texts [J]. Communications in Computer & Information Science, 2014, 49 (6): 103-112.

[11] Zhang X, Wu B. Short text classification based on feature extension using the n-gram model [C]// Proc of International Conference on Fuzzy Systems and Knowledge Discovery. 2016: 710-716.

[12] Bai R, Wang X, Wang X. Literature similarity detection based on digital fingerprint [J]. Library & Information Service, 2013, 3 (3): 128-133.

[13] Wibowo A T, Sudarmadi K W, Barmawi A M. Comparison between fingerprint and winnowing algorithm to detect plagiarism fraud on Bahasa Indonesia documents [C]// Information and Communication Technology. 2013: 128-133.

[14] 朱波，郑虹，孙琳琳. 代码抄袭检测中串匹配算法的比较 [J]. 长春工业大学学报, 2014, 35 (6): 672-676.

[15] Acampora G, Cosma G. A Fuzzy-based approach to programming language independent source-code plagiarism detection [C]// Proc of IEEE International Conference on Fuzzy Systems. 2015: 1-8.

[16] 胡学钢，杨超群，张玉红. 基于自身特征扩展的短文本分类方法 [J]. 计算机应用研究, 2017, 34 (4): 1008-1010.

[17] 刘震，陈晶，郑建宾，等. 中文短文本聚合模型研究 [J]. 软件学报, 2017, 28 (10): 2674-2692.